Programski jezik JavaScript

JavaScript je skriptni ili programski jezik koji omogućava primenu složenih funkcija na web stranicama, kao što su:

- dinamičko prikazivanje informacija;
- ažuriranje sadržaja u realnom vremenu;
- interaktivne mape;
- 2D i 3D animacije...

U ovoj lekciji ćemo proći kroz osnovne korake u JavaScript programskom jeziku, poput kreiranja promenljivih, kreiranja funkcija, prosleđivanja parametara unutar funkcija i ispisivanja rezultata korisniku, kako biste stekli znanje potrebno za uspešno slanje zahteva ka Python serveru, koje će biti obrađeno u jednoj od narednih lekcija.

Napomena:

JavaScript nije potrebno inicijalizovati, jer je već inicijalizovan i aktivan u pozadini pregledača koji koristimo na računaru, tabletu ili telefonu.

Inicijalizacija funkcija i varijabli

Jezik JavaScript nam omogućuje da dinamički menjamo sadržaj internet stranice. Kako bi to bilo izvodljivo, potrebno je znati osnovna pravila. Kroz ovo poglavlje učimo kako možemo definisati promenljive i funkcije.

Postoje tri ključne reči za kreiranje varijabli (promenljivih), a to su var, let i const. Prva ključna reč za definisanje promenljivih, var, definisana je 1997. godine, dok su let i const predstavljene 2015. godine.

Ključna reč var služi za definisanje promenljive. Na primer:

Radno okruženje

```
var name = "Ana";
console.log(name);
```

Objašnjenje:

Rezultat ovog koda biće ispis promenljive name u konzoli pregledača.

U primeru 1 koristili smo funkciju console.log(), koju nam je obezbedio window objekat, koju svaki pregledač ima u sebi. Window objekat predstavlja otvoreni prozor u pregledaču. Funkcije koje nam obezbeđuje window objekat možemo upotrebiti u bilo kom delu JavaScript koda. Više o funkcijama i atributima window objekta moguće je pročitati u zvaničnoj dokumentaciji.

Izmenite u radnom okruženju kod tako da se u konzoli ispište prvo Vaše ime pa prezime.

Napomena:

Detaljno objašnjenje za pokretanje JavaScript koda se nalazi u poglavlju *JavaScript i HTML*.

Ključna reč **let** za kreiranje promenljive je slična reči var. Isto se inicira i isto koristi.

Primer 2

```
let name = "Ana";
console.log(name);
```

Rezultat u konzoli:

```
"Ana"
```

Ključnu reč const koristimo za definisanje promenljivih kad želimo da iniciramo promenljivu koju kasnije ne želimo da menjamo. Ukoliko je promenljiva tipa objekat ili niz, onda ju je moguće menjati, tj. moguće je dodati atribute unutar objekta ili elemente unutar niza i manipulisati njima. Razlog ovakvog ponašanja je taj da ukoliko dodelimo referencu (adresu) promenljivoj koja je definisana pomoću ključne reči **const**, nije moguće menjati adresu ovoj promenljivoj, nego samo dodati vrednost na istu adresu. Izmenom ili dodavanjem sadržaja u nizu ili objektu mi ne menjamo referencu, nego samo unapređujemo već postajeću. U modernim aplikacijama se više koriste varijable let i const. Razlog ovakve pojave je taj da nam ove dve varijable olakšavaju traženje grešaka i njihovo prikazivanje.

Primer 3

```
const name = "Ana";
console.log(name);
```

Rezultat u konzoli:

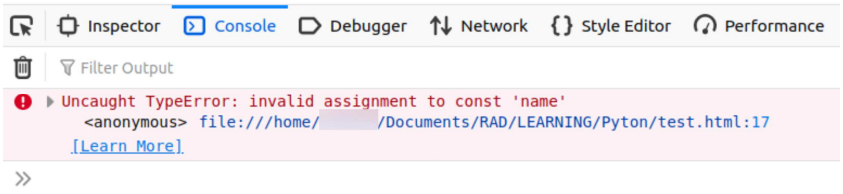
```
"Ana"
```

Izmenite u radnom okruženju kod tako što definišete konstantu pi=3.14 i ispisujete obim kruga za poluprečnik vrednosti 1. Rezultat ispišite sa console.log.

Ukoliko pokušamo da promenimo vrednost const varijable, to će izgledati ovako:

Primer 4

```
const name = "Ana";
name = "Michael";
```



Slika 7.1. Prikaz rezultata primera br. 4

Inicijalizacija funkcije se vrši prema određenim pravilima. Prvo pravilo je da pre naziva funkcije upotrebimo ključnu reč function, nakon koje sledi ime funkcije. Drugo pravilo je da nakon naziva funkcije ide par zagrada (otvorena i zatvorena). Kada se ispoštuju ova pravila, otvaramo prostor u kome pišemo kod koji funkcija treba da izvrši i to radimo sa vitičastim zagradama. Ovaj deo koda se još naziva i telo funkcije. Definisanje funkcije po pravilima izgleda ovako:

```
function nameOfFunction() { ...code }
```

Ulazni parametri u funkcijama

Prilikom pozivanja funkcije moguće je proslediti parametre. Parametri se unose u zagradu prilikom inicijacije funkcije:

```
functionName(param1, param2, param3...)
```

Primer prosleđivanja dva broja i njihovog sabiranja može se videti ispod.

Primer 5

```
function sumOfTwoNumbers(firstNum, firstNum) {
    return (firstNum + firstNum);
};

let sum = sumOfTwoNumbers(5, 7);
console.log(sum)
```

Rezultat:

```
12
```

U primeru možemo videti da je inicijalizovana funkcija pod imenom sumOfTwoNumbers, kojoj su prosleđena dva parametra (5, 7). Nakon inicijalizacije napravljena je promenljiva let i nazvana je sum; njoj je kao referenca postavljena funkcija sumOfTwoNumbers(5, 7). Ovo u ovom trenutku izgleda čudno – zar nije nemoguće dodeliti funkciju promenljivoj?

Ukoliko funkcija ima parametar return, ta funkcija vraća neku vrednost, a ta vrednost može biti na primer niz, broj ili tekst. U našem slučaju je to zbir brojeva 5 i 7, odnosno broj. Kako ta funkcija vraća vrednost, moguće je tu vrednost dodeliti varijabli sum.

Dakle, sad imamo varijablu sum koja ima vrednost 12.

Iskoristite radno okruženje da promenite kod tako što ćete predefinisati funkciju tako da prima 3 cela broja kao parametra i vraća njihov proizvod. Proizvo ispišite u konzoli.

Radno okruženje

JavaScript i HTML

Jedna od funkcija jezika JavaScript je da nam omogućuje manipulaciju HTML koda. JavaScript kod je moguće pisati direktno unutar HTML fajla, ali isto tako ga je moguće pisati i u zasebnom fajlu.

Pisanje JavaScript koda unutar HTML fajla se vrši pomoću posebnog script taga.

Primer 6

```
<script>
     ... some JavaScript code
</script>
```

Ukoliko želimo odvojeno da pišemo JavaScript i HTML kod, koristimo dva fajla. Nazvaćemo ih index.html za pisanje HTML koda i logic.js za pisanje JavaScript koda.

logic.js

```
some JavaScript code
```

U sledećem bloku HTML koda vidimo da je putem atributa unutar script taga dodeljen eksterni link ka našem logic.js fajlu, preko koga povezujemo svoju HTML stranicu sa JavaScript kodom. U ovom slučaju, oba fajla se nalaze unutar istog foldera.

index.html

```
<!DOCTYPE html>
<html lang="en">

<head>
     <title>Document</title>
</head>

<body>
     … some HTML code
// Next line of code is used to add external JS file
 <script src="logic.js"></script>
</body>

</html>
```

Pristup spoljašnjim varijablama i ugrađene funkcije

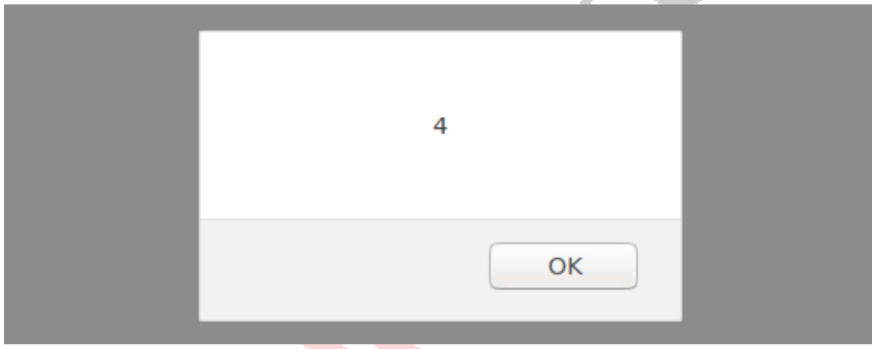
Ugrađene funkcije su funkcije koje sam window objekat već ima u sebi; one se mogu zvati u bilo kom delu JavaScript koda. Neke od njih su alert() i console.log().

Objasnićemo to kroz dva primera:

Primer 7

```
var a = 4;
function f() {
    alert (a);
 }
f();
```

Funkcija alert() je izbacila prozor koji izgleda kao na slici 7.2. Kao rezultat, u prozoru imamo vrednost 4, koju smo dobili globalnom varijablom tipa var koju smo preneli u funkciju f() bez korišćenja ulaznog parametra, jer je varijabla globalna i može joj se pristupiti iz funkcije f().



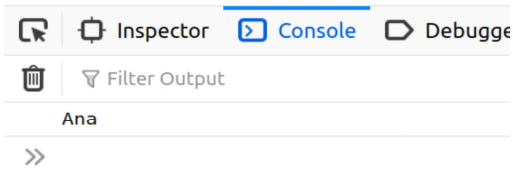
Slika 7.2. Prikaz rezultata primera br. 7

U radnom okruženju izmenite vrednost promenljive a kako bi se unutar prozora ispisala drugačija vrednost.

Primer 8

```
var a = “Ana”;
function f() {
    console.log (a);
}
f();
```

Za testiranje koda možemo ispisivati rezultate unutar konzole. Postoji par načina da proverimo da li je rezultat ispisan u konzoli. Navešćemo dva najpopularnija. Jedan je jednostavnim pritiskom dugmeta F12 na tastaturi unutar pregledača, a drugi je klikom desnog tastera miša na radni prozor pregledača i izborom opcije Inspect Element.



Slika 7.3. Prikaz rezultata primera br. 8

U sledećem primeru biće predstavljeno kako izgleda greška ReferenceError prilikom poziva na ugrađenu funkciju sa prosleđenim parametrom koji nije inicijalizovan.

Primer 9

```
function f() {
    alert (x);
}
f();
```

Kod ovog primera promenljiva x nije definisana pre poziva funkcije i zato se u konzoli prikazuje ReferenceError greška:

Slika 7.4. Prikaz ReferenceError greške u konzoli

Izmenite primer u radnom okruženju tako što ćete promenljivu x predefinisati unutar funkcije f pre ispisa pomoći alert funkcije.

Radno okruženje

Pitanje

Obeležiti ugrađenu funkciju window objekta:

- **alert()**
- f()
- sumFunction()
- sve ponuđeno

Objašnjenje:

Ugrađene funkcije su funkcije koje mogu biti pozvane u bilo kom bloku JavaScript koda; te funkcije nije programer napisao tokom rada na svom projektu/aplikaciji – to su funkcije koje su ugrađene u window objekat.

Rezime

- JavaScript je skriptni ili programski jezik koji nam omogućuje da dinamički ažuriramo sadržaj jedne internet stranice. Takođe je jezik pomoću koga menjamo vrednost, funkciju, oblik… podatka koje posedujemo.
- Postoje tri tipa varijabli koje JavaScript koristi: var, let i const.
- Ugrađene funkcije su funkcije koje nam pruža window objekat i mogu se koristiti i upotrebiti bilo gde u JavaScropt kodu.
- JavaScript kod možemo da unesemo u HTML fajl na dva načina: direktnim unosom u script tag ili preko eksternog linka.